UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/025,270	12/18/2001	Donald Robert Syme	MS1-1419US	5667

22971 7590 03/31/2006

MICROSOFT CORPORATION
ATTN: PATENT GROUP DOCKETING DEPARTMENT
ONE MICROSOFT WAY
REDMOND, WA 98052-6399

EXAMINER	
YIGDALL, MICHAEL J	
ART UNIT	**PAPER NUMBER**
2192	

DATE MAILED: 03/31/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/025,270	SYME ET AL.	
	Examiner	Art Unit	
	Michael J. Yigdall	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *25 January 2006*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-14,25 and 26* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-14,25 and 26* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on January 25, 2006 has been entered. Claims 1-14, 25 and 26 are pending.

Response to Arguments

2. Applicant's arguments have been fully considered but they are not persuasive.

Applicant concludes that "during execution of the program the generic or 'open type' class of Viroli may not determine the current typing context of the object as in claim 1 because the generic or 'open type' class of Viroli has been translated and no longer represents the generic or 'open type' class," and that "the fields in the typing context data structure of Viroli do not describe the exact type" but rather "describe translated objects" (remarks, pages 15-16).

However, notwithstanding Applicant's characterization of the reference (remarks, page 10-16), the plain language of the claims does not exclude Viroli. The feature at issue is recited in the claims as, "the field describing the exact type of the typing-context-relevant-code-point in the typing context" (claim 1, lines 14-15). Viroli teaches this feature. For example, Viroli discloses, "When a client class wants to register a type descriptor, say the one for type Cell<Integer>, it will call a method $TDManager.register passing it the object Integer.class and an array of $TD which contains the type descriptor for class Integer" (page 11, section 4.1, third paragraph). In

this example, "Cell<Integer>" is the exact type of the typing-context-relevant-code-point. In fact, Viroli discloses that "$TD is the class of all the type descriptors" and that it "stores the Class representation of the current type" (page 12, section 4.2, first paragraph). That is, the type descriptors of Viroli describe "exact types" with Class representations of those types.

Applicant refers to the source code illustrated in Figure 10 on page 14 of Viroli (remarks, pages 14-15). Here, class "Cell" is a generic or "open type" class that still represents the generic class even after translation: "public class Cell implements $Parametric." Viroli discloses that the "$Parametric" interface is "automatically implemented by the generic classes, and allows fast retrieval of the type descriptor of an object created by a parametric type" (page 12, section 4.2, first paragraph). Returning to the source code, the line "Cell c1=new Cell($t[1],null)" is a typing-context-relevant-code-point, and the field "$t[1]" describes the exact type of "c1" because "$t[1]" is the type descriptor for "Cell<Integer>." Thus, as noted above, the plain language of the claims does not exclude Viroli.

Response to Amendment

3. It is noted that Applicant has not addressed the rejection of claims 1-14 under 35 U.S.C. 101, as set forth in the Office action mailed on August 25, 2005 and as clarified below.

Claim Rejections - 35 USC § 101

4. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

5. Claims 1-14 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

Claims 1-14 are directed to a computer program product encoding a computer program. However, Applicant defines "computer program product" to include "a computer data signal embodied in a carrier wave" (specification, page 4, first paragraph). Signals and carrier waves do not fall within any class of statutory subject matter, and thus claims 1-14 are not limited to statutory subject matter. See *Interim Guidelines for Examination of Patent Applications for Patent Subject Matter Eligibility* (1300 OG 142), Annex IV.

Claim Rejections - 35 USC § 102

6. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

7. Claims 1-5, 7-9, 25 and 26 are rejected under 35 U.S.C. 102(b) as being anticipated by Viroli et al., *Parametric Polymorphism in Java through the Homogeneous Translation LM: Gathering Type Descriptors at Load-Time* (art of record, "Viroli").

With respect to claim 1 (currently amended), Viroli discloses a computer program product encoding a computer program for executing on a computer system a computer process for dynamically generating typing context data (see, for example, the abstract, and page 6, section 3, which shows a translation to dynamically generate type descriptor objects, and see, for

example, page 8, Figure 4, which shows a type descriptor) associated with a typing-context-relevant-code-point being executed within a typing context in a dynamic execution environment (see, for example, page 11, section 4.1, first paragraph, which shows instantiating parametric types), the computer process comprising:

(a) encountering the typing-context-relevant-code-point in the typing context during execution of the program (see, for example, page 6, section 3, which shows encountering a type operation);

(b) identifying a typing context handle associated with the typing context, the typing context handle referencing a typing context data structure associated with the typing context (see, for example, page 12, section 4.2, first paragraph, which shows a type descriptor manager, type descriptors and hash tables);

(c) computing the typing context data associated with the typing-context-relevant-code-point (see, for example, page 6, section 3, which shows the translation, and page 11, section 4.1, first paragraph, which shows instantiating the parametric types);

(d) dynamically allocating a field in the typing context data structure associated with the typing-context-relevant-code-point, the field describing the exact type of the typing-context-relevant-code-point in the typing context (see, for example, page 11, section 4.1, third paragraph, which shows dynamically allocating a type descriptor that describes an exact type such as "Cell<Integer>," and page 12, section 4.2, first paragraph, which further shows that the type descriptor stores a representation of the exact type); and

(e) recording the typing context data in the field of the typing context data structure (see, for example, page 11, section 4.1, second and third paragraphs, which shows recording the type descriptor in a hash table).

With respect to claim 2 (original), the rejection of claim 1 is incorporated, and Viroli further discloses the limitations wherein the typing-context-relevant-code-point executes a type test on an instance of a generic class (see, for example, page 8, first paragraph, which shows instance tests), the typing context data includes a resource type descriptor defining the exact type of the instance (see, for example, page 8, Figure 4, which shows a type descriptor, and page 11, section 4.1, second and third paragraphs, which shows an array of type descriptors and the exact type of the instance), and the computer process further comprises: performing the type test based on the resource type descriptor associated with the typing-context-relevant-code-point (see, for example, page 8, first paragraph, which shows performing the instance test based on the type).

With respect to claim 3 (original), the rejection of claim 1 is incorporated, and Viroli further discloses the limitations wherein the typing-context-relevant-code-point executes an allocation of an instance of a generic class (see, for example, page 11, section 4.1, first paragraph, which shows instantiating a parametric type), the typing context data includes a resource type descriptor defining the exact type of the instance (see, for example, page 8, Figure 4, which shows a type descriptor, and page 11, section 4.1, second and third paragraphs, which shows an array of type descriptors and the exact type of the instance), and the computer process further comprises: creating the instance of the generic class based on the resource type descriptor associated with the typing-context-relevant-code-point, wherein the instance is of the exact type

(see, for example, page 11, section 4.1, first three paragraphs, which shows creating an instance based on the type, and page 12, last paragraph, second bullet).

With respect to claim 4 (original), the rejection of claim 1 is incorporated, and Viroli further discloses the limitations wherein the typing-context-relevant-code-point calls a generic method, the typing context data includes another typing context handle, and the computer process further comprises: passing the other typing context handle referencing the typing context data to the generic method as a hidden parameter (see, for example, page 12, first paragraph, which shows calling a method for friend types).

With respect to claim 5 (original), the rejection of claim 1 is incorporated, and Viroli further discloses the limitation wherein the identifying operation comprises: retrieving the typing context handle from a stack frame (see, for example, the abstract, which shows a Java virtual machine).

With respect to claim 7 (original), the rejection of claim 1 is incorporated, and Viroli further discloses the limitation wherein the computing operation comprises: retrieving the typing context data associated with the typing-context-relevant-code-point from a global hash table (see, for example, page 11, section 4.1, third paragraph, which shows retrieving the type descriptor from a hash table).

With respect to claim 8 (original), the rejection of claim 1 is incorporated, and Viroli further discloses the limitation wherein the encountering operation comprises: assigning an index

to the typing-context-relevant-code-point (see, for example, page 11, section 4.1, third paragraph, which shows assigning a unique identifier).

With respect to claim 9 (original), the rejection of claim 8 is incorporated, and Viroli further discloses the limitation wherein the allocating operation comprises: allocating the field in the typing context data structure, in accordance with the index (see, for example, page 11, section 4.1, third paragraph, which shows dynamically allocating a type descriptor in accordance with a unique identifier).

With respect to claim 25 (currently amended), the execution engine recited in the claim corresponds to the computer program product of claim 1 (see the rejection of claim 1 above).

With respect to claim 26 (currently amended), the method recited in the claim corresponds to the computer program product of claim 1 (see the rejection of claim 1 above).

Claim Rejections - 35 USC § 103

8. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

9. Claims 6 and 10-14 are rejected under 35 U.S.C. 103(a) as being unpatentable over Viroli, as applied to claim 1 above, in view of U.S. Pat. No. 5,093,914 to Coplien et al. (art of record, "Coplien").

With respect to claim 6 (original), the rejection of claim 1 is incorporated, and Viroli further discloses the limitation wherein the typing-context-relevant-code-point is executed within an instance of a generic class (see, for example, page 11, section 4.1, first paragraph, which shows instantiating a parametric type). Viroli does not expressly disclose the limitation wherein the identifying operation comprises:

(a) retrieving a first pointer to the instance; and

(b) retrieving the typing context handle via a second pointer, a second pointer being relative to the first point and referencing the typing context handle associated with the instance.

However, Coplien teaches step (a) above in terms of a first pointer to a window instance (see, for example, column 10, lines 14-21). Coplien further teaches step (b) above in terms of second pointer to a generic window (see, for example, column 10, lines 23-29) that is relative to the first pointer (see, for example, column 10, lines 31-35) and references type information associated with the instance (see, for example, column 10, lines 37-47).

Coplien is directed to executing parameterized polymorphic code (see, for example, column 10, lines 48-65). It would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teachings of Coplien into Viroli so as to include first and second pointers. The motivation for doing so would have been to facilitate the sharing of base or generic code by specific type instances or subclasses deriving from the same generic or base class or object, thus eliminating redundant code in subclasses which prefer the default semantics defined in the base class.

With respect to claim 10 (original), the rejection of claim 8 is incorporated. Viroli does not expressly disclose the limitation wherein the index is assigned based on the "arity" of the typing-context-relevant-code-point.

However, Coplien discloses this limitation in terms of an index based on the number of arguments (see, for example, column 12, lines 35-42).

Coplien is directed to executing parameterized polymorphic code (see, for example, column 10, lines 48-65). It would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teachings of Coplien into Viroli so as to assign the index based on "arity." The motivation for doing so would have been to facilitate the construction of function signatures (each consists of a function name and the number of arguments), for the purpose of organizing and identifying like-named functions that have different argument lists.

With respect to claim 11 (original), the rejection of claim 8 is incorporated. Viroli does not expressly disclose the limitation wherein the index is assigned based on a category associated with the typing-context-relevant-code-point.

However, Coplien discloses this limitation in terms of an index based on a category such as the number of arguments (see, for example, column 12, lines 35-42).

Coplien is directed to executing parameterized polymorphic code (see, for example, column 10, lines 48-65). It would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teachings of Coplien into Viroli so as to assign the index based on a category. The motivation for doing so would have been to facilitate the construction of function signatures (each consists of a function name and the number of

arguments), for the purpose of organizing and identifying like-named functions that have different argument lists.

With respect to claim 12 (original), the rejection of claim 11 is incorporated, and Coplien further teaches the limitation wherein the category is assigned on a per-containing class basis (see, for example, FIG. 7, which shows an "XWindow" object and a "SunviewWindow" object).

With respect to claim 13 (original), the rejection of claim 11 is incorporated, and Coplien further teaches the limitation wherein the category is assigned on a per-containing method basis (see, for example, FIG. 7, which shows an "XWindow::draw" method and a "SunviewWindow::draw" method).

With respect to claim 14 (original), the rejection of claim 11 is incorporated, and Coplien further teaches the limitation wherein the category is assigned on a per-containing assembly basis (See, for example, FIG. 7, which shows an "XWindow" implementation and a "SunviewWindow" implementation).

Conclusion

10. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Michael J. Yigdall whose telephone number is (571) 272-3707. The examiner can normally be reached on Monday through Friday from 7:30am to 4:00pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

MY

Michael J. Yigdall
Examiner
Art Unit 2192

mjy

Chameli C. Das
CHAMELI C. DAS
PRIMARY EXAMINER

3/21/06